

# Stability of Evolving Multi-Agent Systems

Philippe De Wilde, *Senior Member, IEEE*, and Gerard Briscoe

**Abstract**—A Multi-Agent System is a distributed system where the agents or nodes perform complex functions that cannot be written down in analytic form. Multi-Agent Systems are highly connected, and the information they contain is mostly stored in the connections. When agents update their state, they take into account the state of the other agents, and they have access to those states via the connections. There is also external, user-generated input into the Multi-Agent System. As so much information is stored in the connections, agents are often memory-less. This memory-less property, together with the randomness of the external input, has allowed us to model Multi-Agent Systems using Markov chains. In this paper, we look at Multi-Agent Systems that evolve, i.e. the number of agents varies according to the fitness of the individual agents. We extend our Markov chain model, and define *stability*. This is the start of a methodology to *control* Multi-Agent Systems. We then build upon this to construct an entropy-based definition for the *degree of instability* (entropy of the limit probabilities), which we used to perform a *stability analysis*. We then investigated the stability of evolving agent populations through simulation, and show that the results are consistent with the original definition of stability in non-evolving Multi-Agent Systems, proposed by Chli and De Wilde. This paper forms the theoretical basis for the construction of Digital Business Ecosystems, and applications have been reported elsewhere.

## I. INTRODUCTION

Multi-Agent Systems is a growing field primarily because of recent developments of the Internet as a means of circulating information, goods and services. Many researchers have contributed valuable work to the area in recent years [39]. However, despite both Evolutionary Computing and Multi-Agent Systems being mature research areas [38], [30] their integration, creating evolving agent populations, is a recent development [47]. This integration is non-trivial, because agents can be considered as state machines and Evolutionary Computing algorithms have been developed to work on numerical data and strings without memory effects. So, our aim here is to determine, for Multi-Agent Systems which make use of Evolutionary Computing [29], [7], [46], macroscopic variables that characterise their stability.

While there are several definitions of stability defined for Multi-Agent Systems [40], [1], [34], [54], [33], they are not applicable because of the Evolutionary Computing dynamics inherent in the context of evolving agent populations. Chli and De Wilde [15] model Multi-Agent Systems as Markov chains, which are an established modelling approach in Evolutionary Computing [44]. They model agent evolution in time as Markov processes, and so view a Multi-Agent System as a discrete time Markov chain with potentially unknown

transition probabilities, considered *stable* when its state has converged to an equilibrium distribution [15]. Also, while there is past work on modelling Evolutionary Computing algorithms as Markov chains [43], [36], [24], [20], we have found none including Multi-Agent Systems.

Therefore, we decided to *extend* the *existing* Chli-DeWilde definition of agent stability to include the dynamics of Evolutionary Computing, including *population dynamics* and *macro-states* of the population state-space.

We have applied our efforts reported here to the construction of Digital Business Ecosystems [35], [6], Digital Ecosystems (distributed adaptive open socio-technical systems, with properties of self-organisation, scalability and sustainability, inspired by natural ecosystems) for conducting business that enable network-based economies, levelling the playing field for Small and Medium sized Enterprises (SMEs) [5], [10]. SMEs provide substantial employment and conduct much innovative activity, but struggle in global markets on a far from level playing field, where large companies have distinct advantages [49]. So, we created Digital Ecosystems to be the *digital counterparts of natural ecosystems* [12], [8], which are considered to be robust, self-organising and scalable architectures that can automatically solve complex, dynamic problems [11], [5], [7]. This lead to a novel optimisation technique inspired by natural ecosystems, where the optimisation works at two levels: a first optimisation, migration of agents which are distributed in a decentralised peer-to-peer network, operating continuously in time; this process feeds a second optimisation based on evolutionary computing that operates locally on single peers and is aimed at finding solutions to satisfy locally relevant constraints. So, the local search is improved through this twofold process to yield better local optima faster, as the distributed optimisation provides prior sampling of the search space through computations already performed in other peers with similar constraints. Therefore, our extended Chli-DeWilde stability was invaluable in understanding and developing the evolving agent populations in these Ecosystem-Oriented Architectures (EOAs) [9], because it was important for us to be able to understand, model, and define stability, including the determination of macroscopic variables to characterise the stability, of the order constructing processes within, the evolving agent populations.

## II. MULTI-AGENT SYSTEMS

A *software agent* is a piece of software that acts, for a user in a relationship of *agency*, autonomously in an environment to meet its designed objectives [57]. So, a Multi-Agent System is a system composed of several *software agents*, collectively capable of reaching goals that are difficult to achieve by an individual agent or monolithic system

Philippe De Wilde is with the Intelligent Systems Lab, Department of Computer Science, Heriot Watt University, United Kingdom, pdw@macs.hw.ac.uk.

Gerard Briscoe is with the Systems Research Group, Computer Laboratory, University of Cambridge, United Kingdom, gerard.briscoe@cl.cam.ac.uk.

[57]. Examples of problems which are appropriate to Multi-Agent Systems research include online trading [42], disaster response [45], and modelling social structures [50]. Multi-agent systems are applied in the real world to graphical applications such as computer games and films. They are also used for coordinated defence systems, with other applications including transportation, logistics, as well as in many other fields. It is widely being advocated for use in networking and mobile technologies, to achieve automatic and dynamic load balancing, high scalability, and self-healing networks.

The systems studied by Wiener [56] model information flow between an actor and the environment. Input, state and output are defined, and consequently an evolution equation can be established. In such a system it is possible to distinguish an observational sequence (of the inputs), followed by a decisional sequence of outputs [53].

### III. CHLI-DEWILDE STABILITY

We will now briefly introduce Chli-DeWilde stability for Multi-Agent System and Evolutionary Computing, sufficiently to allow for the derivation of our extensions to Chli-DeWilde stability to include Multi-Agent Systems with Evolutionary Computing. Chli-DeWilde stability was created to provide a clear notion of stability in Multi-Agent Systems [15], because while computer scientists often talk about stable or unstable systems [52], [4], they did so without having a concrete or uniform definition of stability. So, the Chli-DeWilde definition of stability for Multi-Agent Systems was created [15], derived [14] from the notion of stability defined by De Wilde [19], [28], based on the stationary distribution of a stochastic system, making use of discrete-time Markov chains, which we will now introduce<sup>1</sup>.

If we let  $I$  be a *countable set*, in which each  $i \in I$  is called a *state* and  $I$  is called the *state-space*. We can then say that  $\lambda = (\lambda_i : i \in I)$  is a *measure on  $I$*  if  $0 \leq \lambda_i < \infty$  for all  $i \in I$ , and additionally a *distribution* if  $\sum_{i \in I} \lambda_i = 1$  [14]. So, if  $X$  is a *random variable* taking values in  $I$  and we have  $\lambda_i = \Pr(X = i)$ , then  $\lambda$  is the *distribution of  $X$* , and we can say that a matrix  $P = (p_{ij} : i, j \in I)$  is *stochastic* if every row  $(p_{ij} : j \in I)$  is a *distribution* [14], [37]. We can then extend familiar notions of matrix and vector multiplication to cover a general index set  $I$  of potentially infinite size, by defining the multiplication of a matrix by a measure as  $\lambda P$ , which is given by

$$(\lambda P)_i = \sum_{j \in I} \lambda_j p_{ij}. \quad (1)$$

We can now describe the rules for a Markov chain by a definition in terms of the corresponding matrix  $P$  [14], [37].

**Definition 1.** We say that  $(X^t)_{t \geq 0}$  is a Markov chain with initial distribution  $\lambda = (\lambda_i : i \in I)$  and transition matrix  $P = (p_{ij} : i, j \in I)$  if:

- 1)  $\Pr(X^0 = i_0) = \lambda_{i_0}$  and
- 2)  $\Pr(X^{t+1} = i_{t+1} \mid X^0 = i_0, \dots, X^t = i_t) = p_{i_t i_{t+1}}$ .

We abbreviate these two conditions by saying that  $(X^t)_{t \geq 0}$  is Markov( $\lambda, P$ ).

From this first definition the Markov process is *memoryless*<sup>2</sup>, resulting in only the current state of the system being required to describe its subsequent behaviour. We say that a Markov process  $X^0, X^1, \dots, X^t$  has a *stationary distribution* if the probability distribution of  $X^t$  becomes independent of the time  $t$  [15]. So, the following theorem is an easy consequence of the second condition from the first definition.

**Theorem 1.** A discrete-time random process  $(X^t)_{t \geq 0}$  is Markov( $\lambda, P$ ), if and only if for all  $t$  and  $i_0, \dots, i_t$  we have

$$\Pr(X^0 = i_0, \dots, X^t = i_t) = \lambda_{i_0} p_{i_0 i_1} \cdots p_{i_{t-1} i_t}. \quad (2)$$

This first theorem depicts the structure of a Markov chain [14], [37], [16], illustrating the relation with the stochastic matrix  $P$ . The next Theorem shows how the Markov chain evolves in time, again showing the role of the matrix  $P$ .

**Theorem 2.** Let  $(X^t)_{t \geq 0}$  be Markov( $\lambda, P$ ), then for all  $t, s \geq 0$ :

- 1)  $\Pr(X^t = j) = (\lambda P^t)_j$  and
- 2)  $\Pr(X^t = j \mid X^0 = i) = \Pr(X^{t+s} = j \mid X^s = i) = (P^t)_{ij}$ .

For convenience  $(P^t)_{ij}$  can be denoted as  $p_{ij}^{(t)}$ .

Given this second theorem we can define  $p_{ij}^{(t)}$  as the  $t$ -step transition probability from the state  $i$  to  $j$  [14], and we can now introduce the concept of an *invariant distribution* [14], in which we say that  $\lambda$  is invariant if

$$\lambda P = \lambda. \quad (3)$$

The next theorem will link the existence of an *invariant distribution*, which is an algebraic property of the matrix  $P$ , with the probabilistic concept of an *equilibrium distribution*. This only applies to a restricted class of Markov chains, namely, those with *irreducible* and *aperiodic* stochastic matrices. However, there is a multitude of analogous results for other types of Markov chains which we can refer [37], [16], and the following theorem is provided as an indication, of the family of theorems that apply. An *irreducible* matrix  $P$  is one for which, for all  $i, j \in I$  there exist a sufficiently large  $t$  such that  $p_{ij}^{(t)} > 0$ . A matrix  $P$  is *aperiodic* if for

<sup>1</sup>A more comprehensive introduction to Markov chain theory and stochastic processes is available in [37] and [16].

<sup>2</sup>Markov systems with probabilities is a very powerful modelling technique, applicable in large variety of scenarios, and it is common to start memoryless, in which the output probability distribution only depends on the current input. However, there are scenarios in which alternative modelling techniques, like queueing systems, are more suitable, such as when there is asynchronous communications, and to fully characterise the system state at time  $(t)$ , the history of states at  $(t-1)$ ,  $(t-2)$ , ... might also need to be considered.

all states  $i \in I$  we have  $p_{ii}^{(t)} > 0$  for all sufficiently large  $t$  [14], [37], [16]. The meaning of these properties can broadly be explained as follows. An irreducible Markov chain is a chain where all states intercommunicate. For this to happen, there needs to be a non-zero probability to go from any state to any other state. This communication can happen in any number  $t$  of time steps. This leads to the condition  $p_{ij}^{(t)} > 0$  for all  $i$  and  $j$ . An aperiodic Markov chain is a chain where all states are aperiodic. A state is aperiodic if it is not periodic. Finally, a state is periodic if subsequent occupations of this state occur at regular multiples of a time interval. For this to happen,  $p_{ii}^{(t)}$  has to be zero for  $t$  an integral multiple of a number. This leads to the condition  $p_{ii}^{(t)} > 0$  for  $a$ -periodicity. For further explanations, please refer to [16].

**Theorem 3.** *Let  $P$  be irreducible, aperiodic and have an invariant distribution,  $\lambda$  an arbitrary distribution, and suppose that  $(X^t)_{t \geq 0}$  is Markov( $\lambda, P$ ) [14], then*

$$\Pr(X^t = j) \rightarrow p_j^\infty \text{ as } t \rightarrow \infty \text{ for all } j \in I \quad (4)$$

and

$$p_{ij}^{(t)} \rightarrow p_j^\infty \text{ as } t \rightarrow \infty \text{ for all } i, j \in I. \quad (5)$$

We can now view a system  $S$  as a countable set of states  $I$  with implicitly defined transitions between them, and at time  $t$  the state of the system is the random variable  $X^t$ , with the key assumption that  $(X^t)_{t \geq 0}$  is Markov( $\lambda, P$ ) [14], [37], [16].

**Definition 2.** *The system  $S$  is said to be stable when the distribution of its states converge to an equilibrium distribution,*

$$\Pr(X^t = j) \rightarrow p_j^\infty \text{ as } t \rightarrow \infty \text{ for all } j \in I. \quad (6)$$

More intuitively, the system  $S$ , a stochastic process  $X^0, X^1, X^2, \dots$  is *stable* if the probability distribution of  $X^t$  becomes independent of the time index  $t$  for large  $t$  [15]. Most Markov chains with a finite state-space and positive transition probabilities are examples of controllable stable systems, because after an initialisation period they reach a stationary distribution [14].

A Multi-Agent System can be viewed as a system  $S$ , with the system state represented by a finite vector  $\mathbf{X}$ , having dimensions large enough to represent the states of the agents in the system. The state vector will consist of one or more elements for each agent, and a number of elements to define general properties<sup>3</sup> of the system state. Hence there are many more states of the system (different state vectors) than there are agents. We can then model agent *death*, i.e. not being present in the system, by setting the vector elements for that agent to a fixed value that is shared by no other agent, called  $d$  [14].

<sup>3</sup>These general properties are intended to represent properties that are external to the agents, and as such could include the coupling between the agents. However, we would expect such properties, as the coupling between the agents, to be stored within the agents themselves, and so be part of the elements defining the agents.

## IV. INCLUDING EVOLUTION

Having now introduced Chli-DeWilde stability, we will now briefly introduce Evolutionary Computing, sufficiently to allow for the derivation of our extensions to Chli-DeWilde stability to include Multi-Agent Systems with Evolutionary Computing. Evolution is the source of many diverse and creative solutions to problems in nature [17], [22]. However, it can also be useful as a problem-solving tool in artificial systems. Computer scientists and other theoreticians realised that the selection and mutation mechanisms that appear so effective in biological evolution could be abstracted to a computational algorithm [30]. This Evolutionary Computing is now recognised as a sub-field of artificial intelligence (more particularly computational intelligence) that involves combinatorial optimisation problems [3].

### A. Evolutionary Algorithms

Evolutionary algorithms are based upon several fundamental principles from biological evolution, including reproduction, mutation, recombination (crossover), natural selection, and survival of the fittest. As in biological populations, evolution occurs by the repeated application of the above operators [2]. An evolutionary algorithm operates on the collection of individuals making up a population. An *individual*, in the natural world, is an organism with an associated fitness [27]. So, candidate solutions to an optimisation problem play the role of individuals in a population, and a cost (fitness) function determines the environment within which the solutions *live*, analogous to the way the environment selects for the fittest individuals. The number of individuals varies between different implementations and may also vary during the use of an evolutionary algorithm. Each individual possesses some characteristics that are defined through its genotype, its genetic composition, which will be passed onto the descendants of that individual [2]. Processes of mutation (small random changes) and crossover (generation of a new genotype by the combination of components from two individuals) may occur, resulting in new individuals with genotypes differing from the ancestors they will come to replace. These processes iterate, modifying the characteristics of the population [2]. Which members of the population are kept, and used as parents for offspring, depends on the fitness (cost) function of the population. This enables improvement to occur [2], and corresponds to the fitness of an organism in the natural world [27]. Recombination and mutation create the necessary diversity and thereby facilitate novelty, while selection acts as a force increasing quality. Changed pieces of information resulting from recombination and mutation are randomly chosen. Selection operators can be either deterministic, or stochastic. In the latter case, individuals with a higher fitness have a higher chance to be selected than individuals with a lower fitness [2].

### B. Evolving Agent Populations

While the construction of Multi-Agent Systems that make use of Evolutionary Computing differs [47], [7], they universally include populations of agents evolving to provide

desired functionality, i.e. *evolving agent populations*. So, extending Chli-DeWilde stability to the class of Multi-Agent Systems that make use of Evolutionary Computing requires understanding population dynamics and macro-states.

1) *Population Dynamics: Change of notation.* Section III has defined the main properties of Markov chains that we use to model Multi-Agent Systems. We have used the standard notation, as used in [37] and many other textbooks on stochastic processes. This notion uses integers to denote states. These integers are easily used to label the rows and columns of the transition matrix. At the end of Section II, we made the step from Markov chain to multi-agent system. There are many more states than agents, and both ought to be labeled by integers. This turns out to be confusing, and therefore we make a *change of notation* here. We will now label the *agents* with integers  $i$ , and the states by vectors. Each agent  $i$  is in a scalar state  $\xi_i$ . All  $n$  agents together are in an  $n$ -dimensional vector state  $\xi$ . (An agent can be described by multiple numbers, but we group these here into a single scalar  $\xi_i$ , just as multiple binary digits can be read as a single integer.)

The states of the agents are random variables. The actual values of the random variable  $\xi_i$  will be denoted by numbers  $X_i$  or  $Y_i$ . The actual values of the vector of random variables  $\xi$  will be denoted by vectors  $\mathbf{X}$  or  $\mathbf{Y}$ . All the states can vary in time, and time will be denoted by the superscript  $t$ . The symbol for  $t$  is the same as in Section II, but the symbols  $i$  and  $X$  have a new meaning. The symbol  $i$  now denotes the agent, and  $X_i$  is a number describing the state of the agent  $i$ .

The change of notation is essential because each agent is described by a random variable that has the Markov property. The multi-agent system is a network of interacting Markov chains. Without our change of notation, one would need multiple subscripts counting different things.

A Multi-Agent System that makes use of Evolutionary Computing, an evolving agent population, is composed of  $n$  agents, with each agent  $i$  in a state  $\xi_i^t$  at time  $t$ , where  $i = 1, 2, \dots, n$ . The states of the agents are *random variables*, and so the state vector for the Multi-Agent System is a vector of random variables  $\xi^t$ , with the time being discrete,  $t = 0, 1, \dots$ . The interactions among the agents are noisy, and are given by the probability distributions

$$\Pr(X_i|\mathbf{Y}) = \Pr(\xi_i^{t+1} = X_i|\xi^t = \mathbf{Y}), \quad i = 1, \dots, n, \quad (7)$$

where  $X_i$  is a value for the state of agent  $i$ , and  $\mathbf{Y}$  is a value for the state vector of the Multi-Agent System. The probabilities implement a Markov process [51], with the noise caused by mutations. Furthermore, the agents are individually subjected to a *selection pressure* from the environment of the system, which is applied equally to all the agents of the population. So, the probability distributions are statistically independent, and

$$\Pr(\mathbf{X}|\mathbf{Y}) = \prod_{i=1}^n \Pr(\xi_i^{t+1} = X_i|\xi^t = \mathbf{Y}). \quad (8)$$

If the occupation probability of state  $\mathbf{X}$  at time  $t$  is denoted by  $p_{\mathbf{X}}^t$ , then

$$p_{\mathbf{X}}^t = \sum_{\mathbf{Y}} \Pr(\mathbf{X}|\mathbf{Y}) p_{\mathbf{Y}}^{t-1}. \quad (9)$$

This is a discrete time equation used to calculate the evolution of the state occupation probabilities from  $t = 0$ , while equation (8) is the probability of moving from one state to another. The Multi-Agent System is self-stabilising if the limit distribution of the occupation probabilities exists and is non-uniform, i.e.

$$p_{\mathbf{X}}^{\infty} = \lim_{t \rightarrow \infty} p_{\mathbf{X}}^t \quad (10)$$

exists for all states  $\mathbf{X}$ , and there are states  $\mathbf{X}$  and  $\mathbf{Y}$  such that

$$p_{\mathbf{X}}^{\infty} \neq p_{\mathbf{Y}}^{\infty}. \quad (11)$$

These equations imply that some configurations of the system after an extended time will be more likely than others, because the likelihood of their occurrence no longer changes. Such a system is *stable*, because the likelihood of states occurring no longer changes with time, and this is the definition of stability developed in [15]. This is stability in the stochastic sense, because there is some indeterminacy in the future evolution described by the probability distributions. So, even with knowing the initial conditions, there are many directions in which the process might evolve, but still some paths will be more probable than others. Equation (10) is the *probabilistic equivalent* of an *attractor*<sup>4</sup> in a system with deterministic interactions, which we had to extend to a stochastic process because mutation is inherent in Evolutionary Computing.

While the number of agents in the Chli-DeWilde formalism can vary, we require it to vary according to the *selection pressure* acting upon the evolving agent population. We must therefore formally define and extend the definition of *dead* agents, by introducing a new state  $d$  for each agent. If an agent is in this state,  $\xi_i^t = d$ , then it is *dead* and does not affect the state of other agents in the population. If an agent  $i$  has low fitness then that agent will likely die, because

$$\Pr(d|\mathbf{Y}) = \Pr(\xi_i^{t+1} = d|\xi^t = \mathbf{Y}) \quad (12)$$

will be high for all  $\mathbf{Y}$ . Conversely, if an agent has high fitness, then it will likely replicate, becoming a similarly successful agent (mutant), or crossover might occur changing the state of the successful agent and another agent.

2) *Population Macro-States:* The state of the system, an evolving agent population,  $S$  is determined by the collection of agents of which it consists at a specific time  $t$ , which potentially changes as the time increases,  $t+1$ . This collection of agents will have varying fitness values, and so the one with the highest fitness at the current time  $t$  is the *current maximum fitness individual*. For example, an evolving agent population with individuals ranging in fitness between 36.2% and 45.8%, the *current maximum fitness individual (agent)* is the one with a fitness of 45.8%. So, we can define a macro-state  $M$  as a set of states (evolving agent populations) with a common property, here possessing at least one copy of the *current maximum fitness individual*. Therefore, by its definition, each macro-state  $M$  must also have a *maximal state* composed entirely of copies of the *current maximum fitness individual*. If the population size is not fixed (not in nature, can be in evolutionary computing), the state space of the evolving agent

<sup>4</sup>An attractor is a set of states, invariant under the dynamics, towards which neighbouring states asymptotically approach during evolution [55].

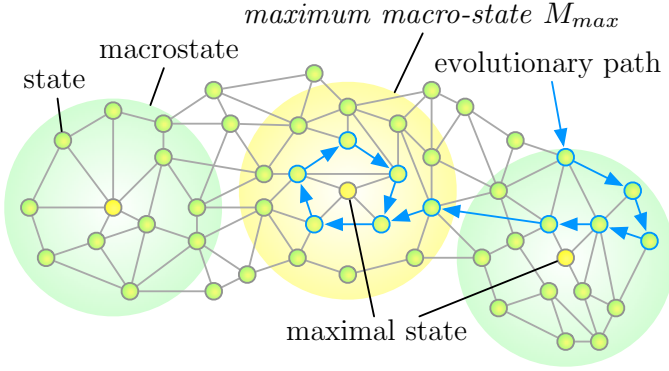


Fig. 1. State-Space of an Evolving Agent Population: A possible evolutionary path through the state-space is shown, with the selection pressure of the evolutionary process driving it towards the maximal state of the maximum macro-state  $M_{max}$ .

population is infinite, but in practise would be bounded by resource availability. So, there is also an infinite number of configurations for an evolving agent population that has the same *current maximum fitness individual*.

So, the state-space  $I$  of the system (evolving agent population)  $S$  can be grouped to a set macro-states  $\{M\}$ . For one macro-state, which we will call the *maximum macro-state*  $M_{max}$ , the *current maximum fitness individual* will be the *global maximum fitness individual*, which is the *optimal solution* (*fittest individual*) that the evolutionary computing process can reach through the evolving agent population (system)  $S$ . For example, an evolving agent population at its *maximum macro-state*  $M_{max}$ , with individuals ranging in fitness between 88.8% and 96.8%, the *global maximum fitness individual* (*agent*) is the one with a fitness of 96.8% and there will be no *fitter* agent. Also, we can therefore refer to all other macro-states of the system  $S$  as *sub-optimal* macro-states, as there can be only one *maximum macro-state*  $M_{max}$ .

We can consider the *macro-states* of an evolving agent population visually through the representation of the state-space  $I$  of the system  $S$  shown in Figure 1, which includes a possible evolutionary path through the state-space. Traversal through the state-space  $I$  is directed by the *selection pressure* of the evolutionary process acting upon the population  $S$ , driving it towards the *maximal state* of the maximum macro-state  $M_{max}$ , consisting entirely of copies of the *optimal solution*. It is the equilibrium state that the system  $S$  is forever falling towards without ever quite reaching, because of the noise (mutation) within the system. Yet, the maximum macro-state  $M_{max}$ , in which this *maximal state* is located, will be reached, provided the system does not get trapped at local optima, i.e. the probability of being in the maximum macro-state  $M_{max}$  at infinite time is one,  $p_{M_{max}}^{\infty} = 1$ .

Furthermore, we can define quantitatively the probability distribution of the macro-states that the system occupies at infinite time. For a stable system, as defined by equation (11), the *degree of instability*,  $\delta$ , is the entropy of its probability distribution at infinite time,

$$\delta = H(p^{\infty}) = - \sum_{\mathbf{x}} p_{\mathbf{x}}^{\infty} \log_N(p_{\mathbf{x}}^{\infty}), \quad (13)$$

where  $N$  is the number of possible states, and taking  $\log$  to the base  $N$  normalises the *degree of instability*. The *degree of instability* will range between zero (inclusive) and one (exclusive), because a maximum instability of one would only occur in the theoretical extreme scenario of a *non-discriminating selection pressure* [25].

## V. SIMULATION AND RESULTS

### A. Evolutionary Dynamics

A simulated agent population was evolved relative to an artificial *selection pressure* created by a *fitness function* generated from a user request  $R$ . An individual (agent) of the population consisted of a set of attributes,  $a_1, a_2, \dots$ , and a user request consisted of a set of required attributes,  $r_1, r_2, \dots$ . The *fitness function* for evaluating an individual agent  $A$ , relative to a user request  $R$ , was

$$fitness(A, R) = \frac{1}{1 + \sum_{r \in R} |r - a|}, \quad (14)$$

where  $a$  is an attribute of the agent  $A$  such that the difference to the required attribute  $r$  of  $R$  was minimised. The abstract agent descriptions was based on existing and emerging technologies for *semantically capable* Service-Oriented Architectures [41], such as the OWL-S semantic markup for web services [31]. We simulated an agent's *semantic description* with an abstract representation consisting of a set of attributes, to simulate the properties of a *semantic description*. Each attribute representing a *property* of the *semantic description*, ranging between one and a hundred. Each simulated agent was initialised with a semantic description of between three and six attributes, which would then evolve in number and content.

Equation 14 was used to assign *fitness* values between 0.0 and 1.0 to each individual of the current generation of the population, directly affecting their ability to replicate into the next generation. The Evolutionary Computing process was encoded with a low mutation rate, a fixed selection pressure and a non-trapping fitness function (i.e. did not get trapped at local optima<sup>5</sup>).

The type of selection used was fitness-proportional and non-elitist, with fitness-proportional meaning that the *fitter* the individual the higher its probability of surviving to the next generation. Non-elitist means that the best individual from one generation was not guaranteed to survive to the next generation; it had a high probability of surviving into the next generation, but it was not guaranteed as it might have been mutated [21]. These initial parameters were chosen to focus

<sup>5</sup>These constraints can be considered in abstract using the metaphor of the *fitness landscape*, in which individuals are represented as solutions to the problem of survival and reproduction [58]. All possible solutions are distributed in a space whose dimensions are the possible properties of individuals. An additional dimension, height, indicates the relative fitness (in terms of survival and reproduction) of each solution. The fitness landscape is envisaged as a rugged, multidimensional landscape of hills, mountains, and valleys, because individuals with certain sets of properties are *fitter* than others [58]. Here the ruggedness of the fitness landscape is not so severe, relative to the population diversity (population size and mutation rate), to prevent the evolving population from progressing to the global optima of the fitness landscape.

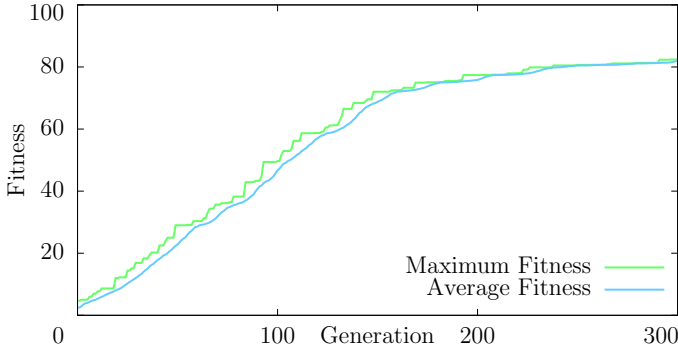


Fig. 2. Graph of Evolutionary Dynamics: This shows both the maximum and average fitness increasing over the generations of a typical evolving agent population, and as expected the average fitness remains below the maximum fitness because of variation in the evolving agent population [23].

on studying our extended Chli-DeWilde stability, rather than the evolutionary computing process itself.

*Crossover* (recombination) was then applied to a randomly chosen 10% of the surviving population. *Mutations* were then applied to a randomly chosen 10% of the surviving population; *point mutations* were randomly located, consisting of *insertions* (an attribute was inserted into an agent), *replacements* (an attribute was replaced in an agent), and *deletions* (an attribute was deleted from an agent). Rates of 10% were chosen, because they would provide the necessary behaviour to a sufficient degree. A dynamic population size was used, with an initial population size of 300, to ensure exploration of the available agent attribute combination space, which increased with the average size of the population's agents, because fixed population sizes can sometimes fail to sufficient search available combination spaces.

The issue of bloat [26] was controlled by augmenting the *fitness function* with a *parsimony pressure* [48], which biased the search to smaller agents, evaluating larger than average agents with a reduced *fitness*, and thereby providing a dynamic control limit which adapted to the average size of the individuals of the evolving agent population.

We first plotted the fitness of the evolutionary process for a typical evolving agent population to elucidate its inherent *evolutionary dynamics*. The graph in Figure 2 shows both the maximum and average fitness increasing over the generations of a typical evolving agent population, and as expected the *average fitness* remains below the *maximum fitness* because of variation in the evolving agent population [23], showing that the inherent dynamism of evolutionary processes applies to evolving agent populations.

## B. Stability

1) *Initial Parameters*: An evolving agent population was called stable if the distribution of the limit probabilities existed and was non-uniform, as defined by equations (10) and (11). The simplest case was a typical evolving agent population with a global optimum, which was stable if there were at least two macro-states with different limit occupation probabilities. So, we considered the maximum macro-state  $M_{max}$  and one of the *sub-optimal macro-states*,  $M_{half}$ . Where the states of the macro-state  $M_{max}$  each possessed at least one individual with

global maximum fitness,

$$p_{M_{max}}^{\infty} = \lim_{t \rightarrow \infty} p_{M_{max}}^{(t)} = 1,$$

while the states of the macro-state  $M_{half}$  each possessed at least one individual with fitness equal to *half* of the global maximum fitness,

$$p_{M_{half}}^{\infty} = \lim_{t \rightarrow \infty} p_{M_{half}}^{(t)} = 0,$$

thereby fulfilling the requirements of equations (10) and (11). A value of  $t = 1000$  was chosen to represent  $t = \infty$  experimentally, because the simulation has often been observed to reach the maximum macro-state  $M_{max}$  within 500 generations. Therefore, the probability of the system  $S$  being in the maximum macro-state  $M_{max}$  at the thousandth generation is expected to be one,  $p_{M_{max}}^{1000} = 1$ . Furthermore, the probability of the system being in the sub-optimal macro-state  $M_{half}$  at the thousandth generation is expected to be zero,  $p_{M_{half}}^{1000} = 0$ .

2) *Predictions*: The sub-optimal macro-state  $M_{half}$ , having a lower fitness, is predicted to be seen earlier in the evolutionary process before disappearing as higher fitness macro-states are reached. The system  $S$  will take longer to reach the maximum macro-state  $M_{max}$ , but once it does will likely remain, leaving only briefly depending on the strength of the mutation rate, as the *selection pressure* was non-elitist.

3) *Results*: Figure 3 shows, for a typical evolving agent population, a graph of the probability as defined by equation (9) of the maximum macro-state  $M_{max}$  and the sub-optimal macro-state  $M_{half}$  at each generation, averaged from ten thousand simulation runs for statistical significance. The behaviour of the simulated system  $S$  was as expected, being in the maximum macro-state  $M_{max}$  only after generation 178 and always after generation 482. It was also observed being in the sub-optimal macro-state  $M_{half}$  only between generations 37 and 113, with a maximum probability of 0.053 at generation 61. This is because the evolutionary path (state transitions) could avoid visiting the macro-state.

4) *Conclusions*: As expected the probability of being in the maximum macro-state  $M_{max}$  at the thousandth generation was one,  $p_{M_{max}}^{1000} = 1$ , and so the probability of being in

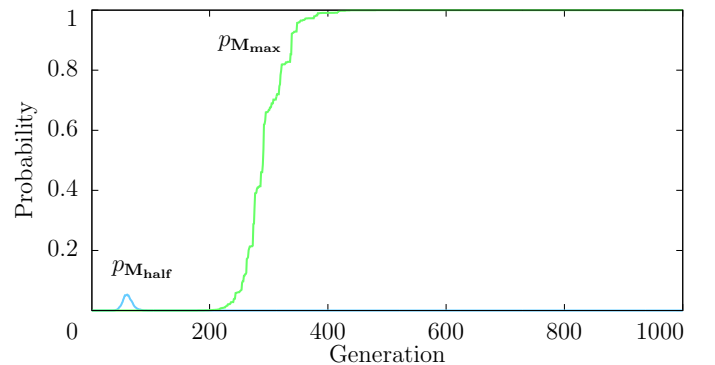


Fig. 3. Graph of the Probabilities of the Macro-States  $M_{max}$  and  $M_{half}$  at each Generation: The system  $S$ , a typical evolving agent population, was in the maximum macro-state  $M_{max}$  only after generation 178 and always after generation 482. It was also observed being in the sub-optimal macro-state  $M_{half}$  only between generations 37 and 113.



any other macro-state, including the sub-optimal macro-state  $M_{half}$ , at the thousandth generation was zero,  $p_{M_{half}}^{1000} = 0$ . We can therefore conclude that our extended Chli-DeWilde stability accurately models the stability over time of evolving agent populations (Multi-Agent Systems with Evolutionary Computing).

### C. Visualisation

1) *Results:* A visualisation for the state of a typical evolving agent population, from the experiment of previous subsection, at the thousandth generation is shown in Figure 4, with each line representing an agent and each shade representing an agent attribute, with the identical agents grouped for clarity. It shows that the evolving agent population reached the maximum macro-state  $M_{max}$  and remained there, but as expected never reached its *maximal state*, where all the agents are identical and have maximum fitness, which is indicated by the lack of total uniformity in the visualisation.

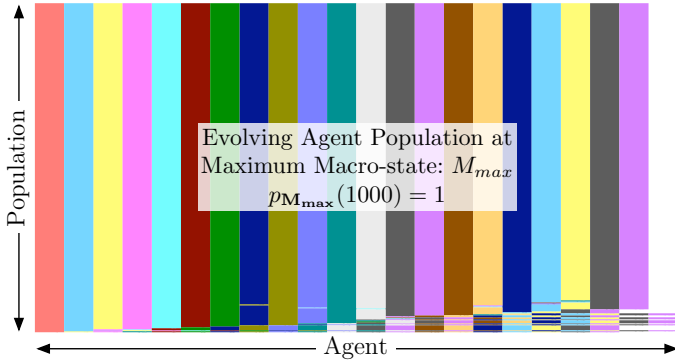


Fig. 4. Visualisation of an Evolving Agent Population at the 1000th Generation: The population consists of 323 agents, with each line representing an agent, and each shade representing an agent attribute. So, we see a population consisting of multiple agents, many of which are identical, having the same maximum global fitness. The identical agents were grouped for clarity, and as expected the system  $S$  reached the maximum macro-state  $M_{max}$ .

2) *Conclusions:* The result was as expected, a lack of total uniformity in the visualisation, because of the mutation (noise) within the evolutionary process, which is necessary to create the opportunity to find fitter (better) sequences and potentially avoid getting trapped at any local optima that may be present. We can therefore conclude that the macro-state interpretation of our extended Chli-DeWilde stability accurately models the state-space of evolving agent populations (Multi-Agent Systems with Evolutionary Computing).

### D. Degree of Instability

1) *Results:* Given that our simulated evolving agent population is stable as defined by equations (10) and (11), we can determine the *degree of instability* as defined by equation (13). So, calculated from its limit probabilities, the *degree of instability* was

$$\begin{aligned} \delta = H(p^{1000}) &= -\sum_{\mathbf{x}} p_{\mathbf{x}}^{1000} \log_N(p_{\mathbf{x}}^{1000}) \\ &= -1 \log_N(1) \\ &= 0, \end{aligned}$$

where  $t = 1000$  was an effective estimate for  $t = \infty$ . This result was expected because the maximum macro-state  $M_{max}$  at the thousandth generation was one,  $p_{M_{max}}^{1000} = 1$ , and so the probability of being in any other macro-state at the thousandth generation was zero.

2) *Conclusions:* The system therefore showed no instability, as there is no entropy in the occupied macro-states at infinite time. We can therefore conclude that the *degree of instability* of our extended Chli-DeWilde stability can provide a macroscopic value to characterise the *level of stability* of evolving agent populations (Multi-Agent Systems with Evolutionary Computing).

### E. Stability Analysis

1) *Initial Parameters:* We then performed a *stability analysis* (similar to a *sensitivity analysis* [13]) of a typical evolving agent population, by varying its key parameters while measuring its stability. We varied the mutation and crossover rates from 0% to 100% in 10% increments to provide a sufficient density of measurements to identify any trends that might be present, calculating the *degree of instability*,  $\delta$  from (13), at the thousandth generation. These *degree of instability* values were averaged over 10 000 simulation runs to ensure statistical significance, and graphed against the mutation and crossover rates in Figure 5.

2) *Results:* It shows that the crossover rate had little effect on the stability of the evolving agent population, whereas the mutation rate did significantly affect stability. With the mutation rate under or equal to 60%, the evolving agent population showed no instability, with  $\delta$  values equal to zero as the system  $S$  was always in the same macro-state  $M$  at infinite time, independent of the crossover rate. With the mutation rate above 60% the instability increased significantly, with the system being in one of several different macro-states at infinite time; with a mutation rate of 70% the system was still very stable, having low  $\delta$  values ranging between 0.08 and 0.16, but

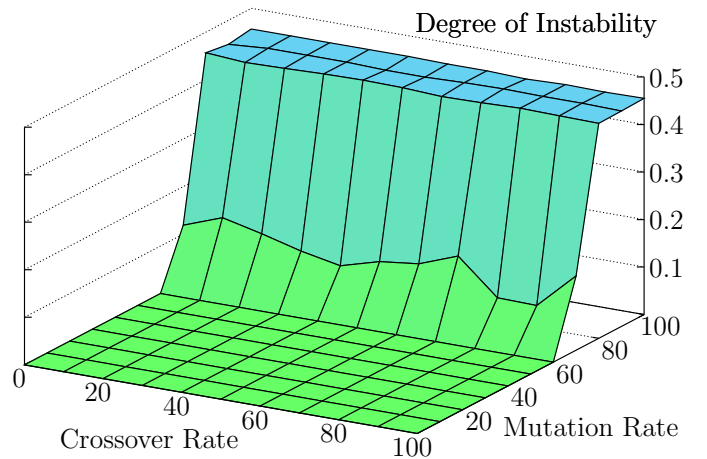


Fig. 5. Graph of Stability with Different Mutation and Crossover Rates: With the mutation rate under or equal to 60%, the evolving agent population showed no instability, with  $\delta$  values equal to zero as the system  $S$  was always in the same macro-state  $M$  at infinite time, independent of the crossover rate. With the mutation rate above 60% the instability increased significantly.

once the mutation rate was 80% or greater the system became quite unstable, shown by high  $\delta$  values nearing 0.5.

3) *Conclusions:* As one would have expected, an extremely high mutation rate had a destabilising affect on the stability of evolving agent populations. Also, as expected the crossover rate had only a minimal effect, because variation from crossover was limited once the population matured, consisting of agents identical or very similar to one another. It should also be noted that the stability of the system is different to its performance at optimising, because while showing no instability with mutation rates below 60% (inclusive), it reached the maximum macro-state  $M_{max}$  only with a mutation rate of 10% or above, while at 0% it was stable at sub-optimal macro-states between  $M_{tenth}$  (inclusive) and  $M_{twentieth}$  (inclusive), i.e. with at least one individual with a fitness between *twentieth* (inclusive) to a *tenth* (inclusive) of the global maximum fitness, as indicated by all of the 0% Mutation Rate experiments have degree of instability,  $\delta$ , values of zero in Figure 5. This is because there was no mutation (mutate rate = 0%), and so the evolving agent populations always remained near the sub-optimal macro-state to which they were initialised (seeded), with any crossover rate having little effect.

We can therefore conclude that the *degree of instability* of our extended Chli-DeWilde stability can be used to perform stability analyses (similar to a *sensitivity analysis* [13]) of evolving agent populations (Multi-Agent Systems with Evolutionary Computing).

## VI. CONCLUSIONS

Our extension of Chli-DeWilde stability was developed to provide a greater understanding of stability in Multi-Agent Systems that make use of Evolutionary Computing, i.e. evolving agent populations. We then built upon this to construct an entropy-based definition for the *degree of instability*, which provides information about the level of stability, applicable to Multi-Agent Systems with or without Evolutionary Computing. Furthermore, it can be used to perform a *stability analysis*, similar to a *sensitivity analysis*, of Multi-Agent Systems.

Collectively, the experimental results confirm that Chli-DeWilde stability has been successfully extended to evolving agent populations, while our definition for the *degree of instability* provides a macroscopic value to characterise the *level of stability*. These findings also support the proposition that Chli-DeWilde stability can be widely applied to different classes of Multi-Agent Systems. So, our extended Chli-DeWilde stability is a useful tool for analysing Multi-Agent Systems, with or without Evolutionary Computing, providing an effective understanding and quantification to help better understand the stability of such systems.

Overall, an insight has been achieved into the stability of Multi-Agent Systems that make use of Evolutionary Computing, which is a first step in being able to control such systems. For example, say one wanted to avoid a number of *bad* states. If the probability of being in those states kept changing with time, it would be difficult to devise a strategy to avoid these

states. However, if the probability converges in time, while one could not guarantee to avoid those states, one could at least calculate the expected damage, i.e. the probability of being in a state times by the penalty for being in it, summed over all the states one wishes to avoid. Stochastic control of multi-agent systems will be the subject of further work.

Our future work will also consider include more experimental scenarios to further consolidate the conclusions, with a range of different *fitness landscapes* [58], including flat ones (from the neutral theory of molecular evolution [25]) and ones with multiple global optima.

## VII. ACKNOWLEDGMENTS

The authors would like to thank the following for encouragement and suggestions; Dr Paolo Dini of the London School of Economics and Political Science, Dr Maria Chli of Aston University, and Dr Jon Rowe of the University of Birmingham. This work was supported by the EU-funded OPAALS Network of Excellence (NoE), Contract No. FP6/IST-034824.

## REFERENCES

- [1] D. Angeli and P.A. Bliman. Stability of leaderless discrete-time multi-agent systems. *Mathematics of Control, Signals, and Systems (MCSS)*, 18(4):293–322, 2006.
- [2] T. Bäck. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press, 1996.
- [3] T. Baeck, D. Fogel, and Z. Michalewicz, editors. *Handbook of Evolutionary Computation*. CRC Press, 1997.
- [4] H. Balakrishnan, M. Stemm, S. Seshan, and R. Katz. Analyzing stability in wide-area network performance. In Scott Leutenegger, editor, *International Conference on Measurement and Modeling of Computer Systems*, pages 2–12. ACM Press, 1997.
- [5] G. Briscoe. *Digital Ecosystems*. PhD thesis, Imperial College London, 2009.
- [6] G. Briscoe. Complex adaptive digital ecosystems. In *ACM Management of Emergent Digital Ecosystems Conference*, 2010.
- [7] G. Briscoe and P. De Wilde. Digital Ecosystems: Evolving service-oriented architectures. In *IEEE Bio Inspired Models of Network, Information and Computing Systems Conference*, 2006.
- [8] G. Briscoe and P. De Wilde. Computing of applied digital ecosystems. In *ACM Management of Emergent Digital Ecosystems Conference*, 2009.
- [9] G. Briscoe and P. De Wilde. Digital Ecosystems: Stability of evolving agent populations. In *ACM Management of Emergent Digital Ecosystems Conference*, 2009.
- [10] G. Briscoe and P. De Wilde. The computing of digital ecosystems. *International Journal of Organizational and Collective Intelligence*, 1(4), 2010.
- [11] G. Briscoe and S. Sadedin. Natural science paradigms. In *Digital Business Ecosystems*, pages 48–55. European Commission, 2007.
- [12] G. Briscoe, S. Sadedin, and G. Paperin. Biology of applied digital ecosystems. In *IEEE Digital Ecosystems and Technologies Conference*, pages 458–463, 2007.
- [13] Dan Cacuci, Mihaela Ionescu-Bujor, and Ionel Navon. *Sensitivity and Uncertainty Analysis*. CRC Press, 2003.
- [14] M. Chli. *Convergence and Interactivity of Multi-Agent Systems*. PhD thesis, Imperial College London, 2006.
- [15] M. Chli, P. De Wilde, J. Goossenaerts, V. Abramov, N. Szirbik, L. Correia, P. Mariano, and R. Ribeiro. Stability of multi-agent systems. In E. Santos Jr and P. Willett, editors, *International Conference on Systems, Man, and Cybernetics*, pages 551–556. IEEE Press, 2003.
- [16] D. Cox and H. Miller. *The Theory of Stochastic Processes*. CRC Press, 1977.
- [17] C. Darwin. *On the Origin of Species by Means of Natural Selection*. John Murray, 1859.
- [18] K. De Jong, D. Fogel, and H. Schwefel. A history of evolutionary computation. In Baeck et al. [3], pages 1–12.

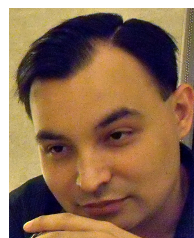


- [19] P. De Wilde, H. Nwana, and L. Lee. Stability, fairness and scalability of multi-agent systems. *International Journal of Knowledge-Based Intelligent Engineering Systems*, 3:84–91, 1999.
- [20] A. Eiben, E. Aarts, and K. Van Hee. Global convergence of genetic algorithms: A Markov chain analysis. In H. Schwefel and R. Manner, editors, *Parallel Problem Solving from Nature*, pages 4–12. Springer, 1991.
- [21] A. Eiben and J. Smith. *Introduction to Evolutionary Computing*. Springer, 2003.
- [22] D. Futuyma. *Evolutionary Biology*. Sinauer Associates, 1998.
- [23] D. Goldberg. *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley, 1989.
- [24] D. Goldberg and P. Segrest. Finite Markov chain analysis of genetic algorithms. In John Grefenstette, editor, *International Conference on Genetic Algorithms and their application*, pages 1–8. Lawrence Erlbaum Associates, 1987.
- [25] M. Kimura. *The neutral theory of molecular evolution*. Cambridge University Press, 1983.
- [26] J. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.
- [27] E. Lawrence. *Henderson's dictionary of biological terms*. Pearson Education, 2005.
- [28] LC Lee, HS Nwana, DT Ndumu, and P. De Wilde. The stability, scalability and performance of multi-agent systems. *BT Technology Journal*, 16:94–103, 1998.
- [29] S. Mabu, K. Hirasawa, and J. Hu. A graph-based evolutionary algorithm: Genetic network programming (GNP) and its extension using reinforcement learning. *Evolutionary Computation*, 15:369–398, 2007.
- [30] P. Marrow. Nature-inspired computing technology and applications. *BT Technology Journal*, 18:13–23, 2000.
- [31] D. Martin, M. Paolucci, S. McIlraith, M. Burstein, D. McDermott, D. McGuinness, B. Parsia, T. Payne, M. Sabou, M. Solanki, Naveen Srinivasan, and Katia Sycara. Bringing semantics to web services: The OWL-S approach. In Jorge Cardoso and Amit Sheth, editors, *Semantic Web Services and Web Process Composition*, pages 6–9. Springer, 2004.
- [32] T. Marwala, P. De Wilde, L. Correia, P. Mariano, R. Ribeiro, V. Abramov, N. Szirbik, and J. Goossenaerts. Scalability and optimisation of a committee of agents using genetic algorithm. In D. Campbell and C. Fyfe, editors, *International ICSC Symposium Soft Computing and Intelligent Systems For Industry*. ICSC-NAISO Academic Press, 2001.
- [33] G. Mohanarajah and T. Hayakawa. Formation stability of multi-agent systems with limited information. In *American Control Conference, 2008*, pages 704–709, 2008.
- [34] L. Moreau. Stability of multiagent systems with time-dependent communication links. *IEEE Transactions on Automatic Control*, 50:169–182, 2005.
- [35] F. Nachira, A. Nicolai, P. Dini, M. Le Louarn, and L. Rivera León, editors. *Digital Business Ecosystems*. European Commission, 2007.
- [36] A. Nix and M. Vose. Modeling genetic algorithms with Markov chains. *Annals of Mathematics and Artificial Intelligence*, 5:79–88, 1992.
- [37] J. Norris. *Markov Chains*. Cambridge University Press, 1997.
- [38] H. Nwana. Software agents: An overview. *Knowledge Engineering Review*, 11:205–244, 1996.
- [39] H.S. Nwana. Software agents: An overview. *Knowledge Engineering Review*, 11(3):205–244, 1996.
- [40] R. Olfati-Saber, J. Fax, and R. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95:215–233, 2007.
- [41] P. Rajasekaran, J. Miller, K. Verma, and A. Sheth. Enhancing web services description and discovery to facilitate composition. In Jorge Cardoso and Amit Sheth, editors, *Semantic Web Services and Web Process Composition*, pages 55–68. Springer, 2004.
- [42] A. Rogers, E. David, N.R. Jennings, and J. Schiff. The effects of proxy bidding and minimum bid increments within eBay auctions. *ACM Transactions on the Web (TWEB)*, 1(2):9, 2007.
- [43] G. Rudolph. Convergence analysis of canonical genetic algorithms. *IEEE Transactions on Neural Networks*, 5:96–101, 1994.
- [44] G. Rudolph. Finite Markov chain results in evolutionary computation: A tour d'horizon. *Fundamenta Informaticae*, 35:67–89, 1998.
- [45] N. Schurr, J. Marecki, M. Tambe, P. Scerri, N. Kasinadhuni, and J. Lewis. The future of disaster response: Humans working with multiagent teams using DEFECTO. In *AAAI Spring Symposium on Homeland Security*, 2005.
- [46] R. Smith, C. Bonacina, P. Kearney, and W. Merlat. Embodiment of evolutionary computation in general agents. *Evolutionary Computation*, 8:475–493, 2000.
- [47] R. Smith and N. Taylor. A framework for evolutionary computation in agent-based systems. In Janice Glasgow, editor, *International Conference on Intelligent Systems*, pages 221–224. AAAI Press, 1998.
- [48] T. Soule and J. Foster. Effects of code growth and parsimony pressure on populations in genetic programming. *Evolutionary Computation*, 6:293–309, 1998.
- [49] J. Stanley and G. Briscoe. The abc of digital business ecosystems. *Communications Law - Journal of Computer, Media and Telecommunications Law*, 15(1), 2010.
- [50] R. Sun and I. Naveh. Simulating organizational decision-making using a cognitively realistic agent model. *Journal of Artificial Societies and Social Simulation*, 7(3), 2004.
- [51] J. Suzuki. A Markov chain analysis on simple genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 25:655–659, 1995.
- [52] James Thomas and Katia Sycara. Heterogeneity, stability, and efficiency in distributed systems. In Yves Demazeau, editor, *International Conference on Multi Agent Systems*, pages 293 – 300. IEEE Press, 1998.
- [53] R. Vallee. Cognition et système (cognition and systems). *Paris: l'Interdisciplinaire Systeme (s)*, 1995.
- [54] G. Weiss. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, 1999.
- [55] E. Weisstein. *CRC Concise Encyclopedia of Mathematics*. CRC Press, 2003.
- [56] N. Wiener. *Cybernetics or Control and Communication in the Animal and the Machine*. MIT Press, 1948.
- [57] M. Wooldridge. *Introduction to MultiAgent Systems*. Wiley, 2002.
- [58] S. Wright. The roles of mutation, inbreeding, crossbreeding and selection in evolution. In D. Jones, editor, *International Congress on Genetics*, pages 356–366. Brooklyn botanic garden, 1932.

## BIOGRAPHIES



**Philippe De Wilde** is a Professor at the Intelligent Systems Lab, Department of Computer Science, and Head of the School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh, United Kingdom. Research interests: stability, scalability and evolution of multi-agent systems; networked populations; coordination mechanisms for populations; group decision making under uncertainty; neural networks, neuro-economics. He tries to discover biological and sociological principles that can improve the design of decision making and of networks. Research Fellow, British Telecom, 1994. Laureate, Royal Academy of Sciences, Letters and Fine Arts of Belgium, 1988. Senior Member of IEEE, Member of IEEE Computational Intelligence Society and Systems, Man and Cybernetics Society, ACM, and British Computer Society. Associate Editor, IEEE Transactions on Systems, Man, and Cybernetics, Part B, Cybernetics.



**Gerard Briscoe** is a Research Associate at the Systems Research Group of the Computer Laboratory, University of Cambridge, UK, and a Visiting Scholar at Intelligent Systems Lab of the School of Mathematical and Computer Sciences, Heriot-Watt University, UK. Before this he was a postdoctoral researcher at the Department of Media and Communications of the London School of Economics and Political Science, UK. He received his PhD in Electrical and Electronic Engineering from Imperial College London, UK. Before which he worked as a Research Fellow at the MIT Media Lab Europe, after completing his B/MEng in Computing also from Imperial College London. His research interests include Sustainable Computing, Cloud Computing, Social Media and Natural Computing.